Homework 6: Threads and Locking

제출마감 : 2018.11.8. 11:00am

제출처 : 오준택 조교(na94jun@gmail.com)

이번 과제를 통해, 쓰레드들을 이용한 병렬 프로그래밍과 lock의 사용법을 익히게 될 것이다. 이번 과제는 xv6 와 qemu 를 사용하지 않고 multi core를 수행할 수 있는 컴퓨터에서 수행해야한다.

piazza에서 *ph.c* 를 다운로드 하고 컴파일하라. ph.c는 해쉬 테이블을 사용하는 예제 코드이다. put () 과 get ()을 통해 해쉬 테이블에 데이터를 삽입하고 얻어오는 동작을 한다.

```
$ gcc -g -O2 ph.c -pthread

$ ./a.out 2
```

위의 두개의 명령을 실행하라. a.out에 전달되는 인자 숫자 2는 실행파일을 수행하는 thread 개수가 2개임을 명시한다.

아래의 출력 결과는 프로그램을 수행했을 때 결과이다.

```
0: put time = 2.871728
1: put time = 2.957073
1: get time = 12.731078
1: 1 keys missing
0: get time = 12.731874
0: 1 keys missing
completion time = 15.689165
```

각 thread는 두 가지 단계로 동작한다. 첫 번째 단계에서 각 thread는 주어진 key의 개수를 쓰레드의 개수로 나눠 (NKEYS/nthread) key들을 해쉬테이블에 삽입한다. 두 번째 단계에서, 각 thread는 주어진 key의 개수 (NKEYS) 만큼 해쉬테이블에서 가져온다. 출력된 결과는 각 thread가 각 단계에서 얼마나 시간을 소요하였는지를 나타낸다. 마지막의 completion time은 실행파일의 총 수행시간을 의미한다. 위의 출력결과에서는 completion time은 약 16초이다. 각각의 thread는 약 16초 (put : 약 3초 , get : 약 13초) 동안 수행된다.

2개의 thread를 사용할 경우 성능향상을 판단하기 위해, 1개의 thread 수행 시 성능을 출력해 비교해보자.

```
$ ./a.out 1
0: put time = 5.350298
0: get time = 11.690395
0: 0 keys missing
completion time = 17.040894
```

1개의 thread 를 수행할 경우 completion time은 약 17초이다. 2개의 thread를 동작하는 경우 (약 15.6초) 보다 약간 더 걸린다. 하지만 2개의 thread일 경우 1개의 thread에 비해 get 수행 동작의 속도가 거의 2배 정도가된다. Put의 수행동작의 경우 속도향상이 되지만 완벽하진 않다; 즉, 두 개의 쓰레드로 돌리는 경우, 삽입 속도가 거의 두 배 가까이 빨라진다.

(두 개의 쓰레드를 돌리는 경우, 삽입 속도가 증가하는 이유는 두 개의 쓰레드가 동시에 동작하기 때문인데, 여러분의 컴퓨터가 오직 하나의 core만 가지고 있거나 컴퓨터가 다른 application들을 처리한다면 성능 향상을 볼 수 없다.)

실행 속도와 무관하게, 여러분은 ph.c 코드가 잘못됐다는 것을 확인할 수 있다. 2개의 thread들이 동작할 때, 실행파일은 첫 번째 단계에서 삽입 된 1개의 key를 두 번째 단계에서 찾을 수 없는 문제가 있다. 출력결과는 4개의 코어에서 4개의 thread들을 동작시킨 결과이다.

```
2: put time = 1.516581
1: put time = 1.529754
0: put time = 1.816878
3: put time = 2.113230
2: get time = 15.635937
2: 21 keys missing
3: get time = 15.694796
3: 21 keys missing
1: get time = 15.714341
1: 21 keys missing
0: get time = 15.746386
0: 21 keys missing
completion time = 17.866878
```

2가지 핵심은 다음과 같다.

1) 2개의 쓰레드의 completion time은 1개의 쓰레드의 결과와 동일하지만 get 동작을 수행할 때 2배의 양을 수행한다

2) 많은 key들을 손실한다. 실행파일 수행시, 많거나 적거나 key들을 손실한다. 일부 실행에서는 손실이 없을 수도 있다. 만약 여러분이 1개의

쓰레드로 실행파일을 수행한다면, key들의 손실을 볼 수 없을 것이다. 2개의 쓰레드를 수행할 때 key들이 손실 될 수 있는 문제를 확인해 보자.

위의 문제를 해결하기 위해, lock과 unlock을 put 과 get에 삽입하라. 그 결과 손실되는 key들이 하나도 존재하지 않게 하라. lock 동작과 관련된 pthread calls은 다음과 같다 : (더 자세히 알고 싶다면, manual page 또는 man명령어를 이용하라)

```
pthread_mutex_t lock;     // declare a lock
pthread_mutex_init(&lock, NULL);   // initialize the lock
pthread_mutex_lock(&lock);  // acquire lock
pthread_mutex_unlock(&lock);  // release lock
```

우선, 1개의 thread로 실행파일을 수행하라. 이후 2개의 thread들로 실행파일을 수행하라. 손실 되는 키들이 존재하는가? 2개의 thread들이 동작하는 실행파일이 1개의 thread로 동작하는 실행파일 보다 빠른가?

get 수행 동작이 손실되는 키 없이 병행적으로 잘 동작하도록 수정하라. (힌트: 현재 실행파일에서 손실되는 키없이 동작하기 위해 get 수행 동작에 lock이 필수적인지 판단하라.)

put 수행 동작이 손실되는 키 없이 병행적으로 잘 동작하도록 수정하라. (Hint: 버킷 하나당 락을 해야할까요?)

Submit: 수정한 ph.c 파일

제출양식: hw6_자신의학번.c